Chapter 4 : Intermediate SQL

Database System Concepts, 7th Ed.

Outline

- Join Expressions
- Views
- Integrity Constraints
- SQL Data Types and Schemas
- Authorization

Joined Relations

- **Join operations** take two relations and return as a result another relation.
- A join operation is a Cartesian product which requires that tuples in the two relations match (under some condition). It also specifies the attributes that are present in the result of the join
- The join operations are typically used as subquery expressions in the **from** clause
- Three types of joins:
  - Natural join
  - Inner join
  - Outer join

Natural Join in SQL

- Natural join matches tuples with the same values for all common attributes, and retains only one copy of each common column.
- List the names of instructors along with the course ID of the courses that they taught

  -
    ```
    select name, course_id
    from  students, takes
    where student.ID = takes.ID;
    ```
- Same query in SQL with “natural join” construct

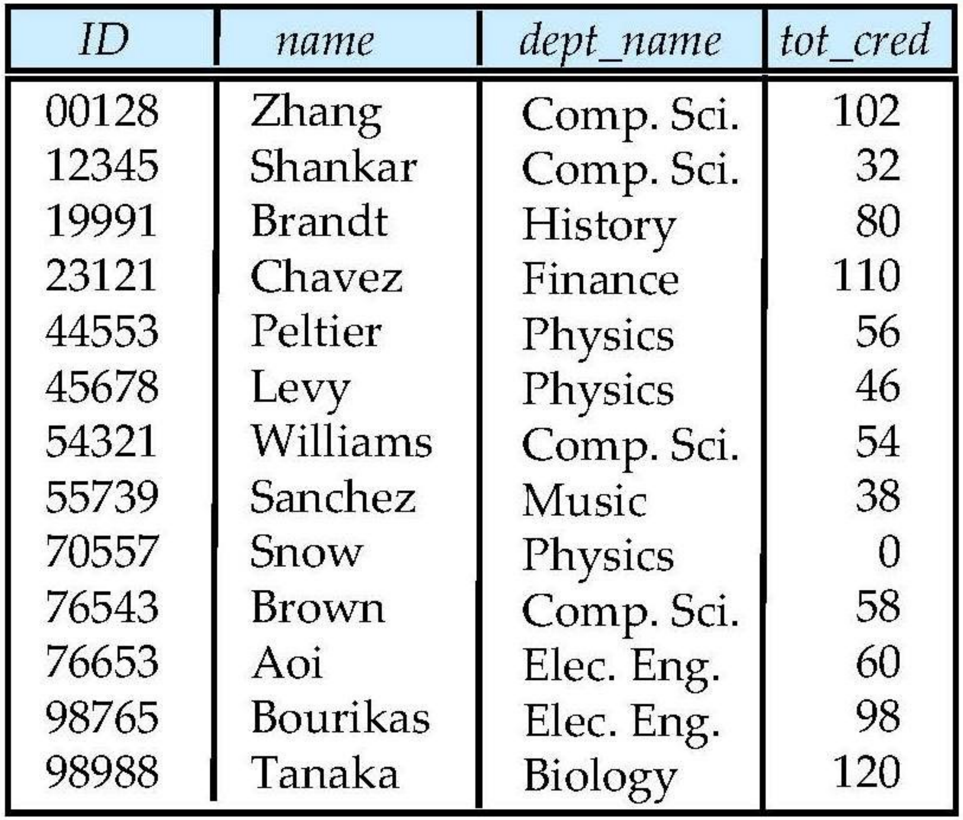
  -
    ```
    select name, course_id
    from student natural join takes;
    ```

Natural Join in SQL (Cont.)

- The **from** clause can have multiple relations combined using natural join:

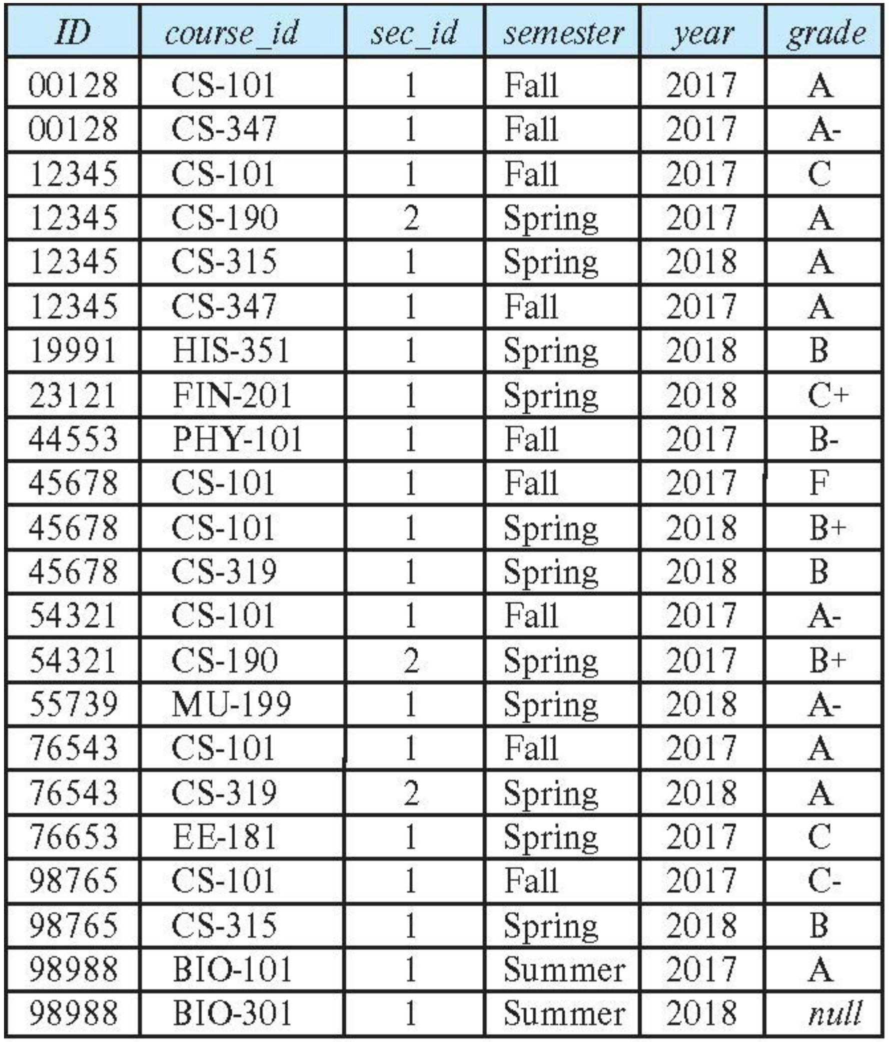
  select $A_1, A_2, \dots A_n$
  from r_1 **natural join** r_2 **natural join .. natural join** r_n
  where P ;

Student Relation



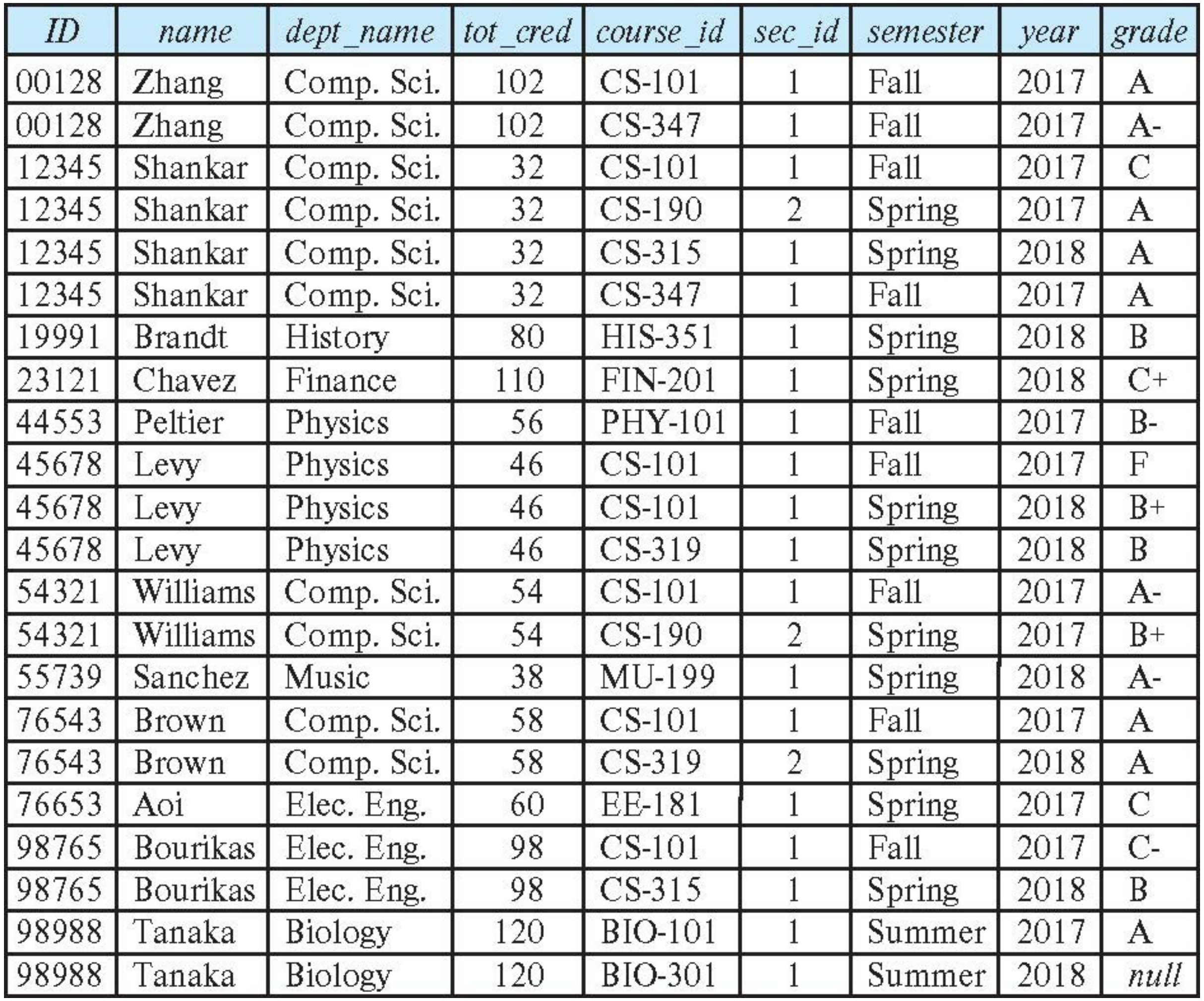
ID	*name*	*dept_name*	*tot_cred*
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120

Takes Relation



ID	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	CS-101	1	Fall	2017	A
00128	CS-347	1	Fall	2017	A-
12345	CS-101	1	Fall	2017	C
12345	CS-190	2	Spring	2017	A
12345	CS-315	1	Spring	2018	A
12345	CS-347	1	Fall	2017	A
19991	HIS-351	1	Spring	2018	B
23121	FIN-201	1	Spring	2018	C+
44553	PHY-101	1	Fall	2017	B-
45678	CS-101	1	Fall	2017	F
45678	CS-101	1	Spring	2018	B+
45678	CS-319	1	Spring	2018	B
54321	CS-101	1	Fall	2017	A-
54321	CS-190	2	Spring	2017	B+
55739	MU-199	1	Spring	2018	A-
76543	CS-101	1	Fall	2017	A
76543	CS-319	2	Spring	2018	A
76653	EE-181	1	Spring	2017	C
98765	CS-101	1	Fall	2017	C-
98765	CS-315	1	Spring	2018	B
98988	BIO-101	1	Summer	2017	A
98988	BIO-301	1	Summer	2018	*null*

student **natural join** *takes*



ID	*name*	*dept_name*	*tot_cred*	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	Zhang	Comp. Sci.	102	CS-101	1	Fall	2017	A
00128	Zhang	Comp. Sci.	102	CS-347	1	Fall	2017	A-
12345	Shankar	Comp. Sci.	32	CS-101	1	Fall	2017	C
12345	Shankar	Comp. Sci.	32	CS-190	2	Spring	2017	A
12345	Shankar	Comp. Sci.	32	CS-315	1	Spring	2018	A
12345	Shankar	Comp. Sci.	32	CS-347	1	Fall	2017	A
19991	Brandt	History	80	HIS-351	1	Spring	2018	B
23121	Chavez	Finance	110	FIN-201	1	Spring	2018	C+
44553	Peltier	Physics	56	PHY-101	1	Fall	2017	B-
45678	Levy	Physics	46	CS-101	1	Fall	2017	F
45678	Levy	Physics	46	CS-101	1	Spring	2018	B+
45678	Levy	Physics	46	CS-319	1	Spring	2018	B
54321	Williams	Comp. Sci.	54	CS-101	1	Fall	2017	A-
54321	Williams	Comp. Sci.	54	CS-190	2	Spring	2017	B+
55739	Sanchez	Music	38	MU-199	1	Spring	2018	A-
76543	Brown	Comp. Sci.	58	CS-101	1	Fall	2017	A
76543	Brown	Comp. Sci.	58	CS-319	2	Spring	2018	A
76653	Aoi	Elec. Eng.	60	EE-181	1	Spring	2017	C
98765	Bourikas	Elec. Eng.	98	CS-101	1	Fall	2017	C-
98765	Bourikas	Elec. Eng.	98	CS-315	1	Spring	2018	B
98988	Tanaka	Biology	120	BIO-101	1	Summer	2017	A
98988	Tanaka	Biology	120	BIO-301	1	Summer	2018	*null*

Dangerous in Natural Join

- Beware of unrelated attributes with same name which get equated incorrectly
- Example -- List the names of students instructors along with the titles of courses that they have taken
  - Correct version

    ```
    select name, title
     from student natural join takes, course
     where takes.course_id = course.course_id;
    ```

  - Incorrect version

    ```
    select name, title
    from student natural join takes natural join course;
    ```

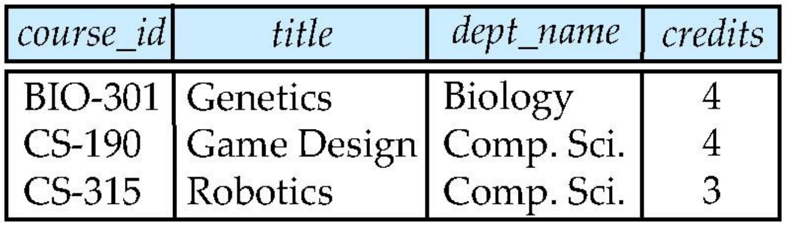
    - This query omits all (student name, course title) pairs where the student takes a course in a department other than the student's own department.
    - The correct version (above), correctly outputs such pairs.

Outer Join

- An extension of the join operation that avoids loss of information.
- Computes the join and then adds tuples form one relation that does not match tuples in the other relation to the result of the join.
- Uses *null* values.
- Three forms of outer join:
  - left outer join
  - right outer join
  - full outer join

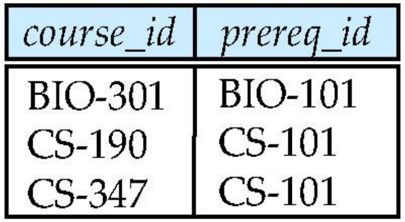
Outer Join Examples

- Relation *course*



course_id	*title*	*dept_name*	*credits*
BIO-301	Genetics	Biology	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3

- Relation *prereq*



course_id	*prereq_id*
BIO-301	BIO-101
CS-190	CS-101
CS-347	CS-101

- Observe that

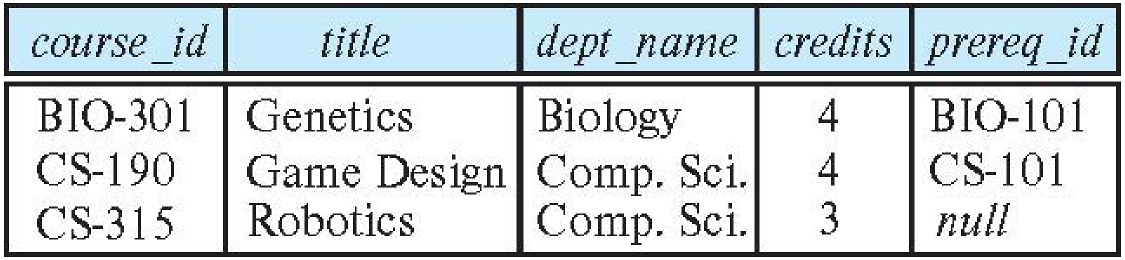
  course information is missing CS-437

  prereq information is missing CS-315

- *x*

Left Outer Join

- *course* **natural left outer join** *prereq*

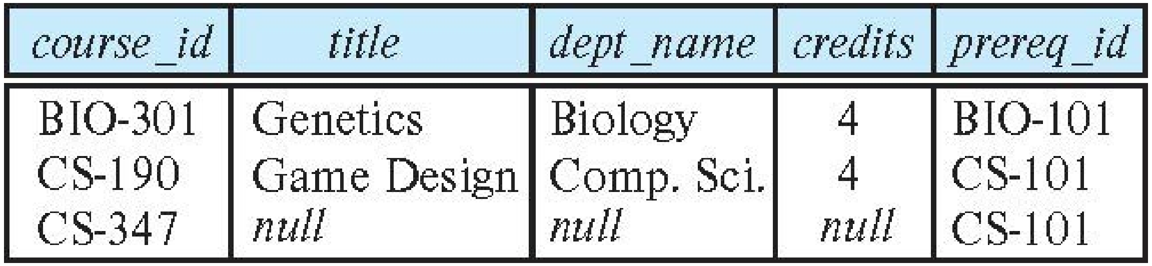


course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	*null*

- In relational algebra: *course* ⟕ *prereq*

Right Outer Join

- *course* **natural right outer join** *prereq*

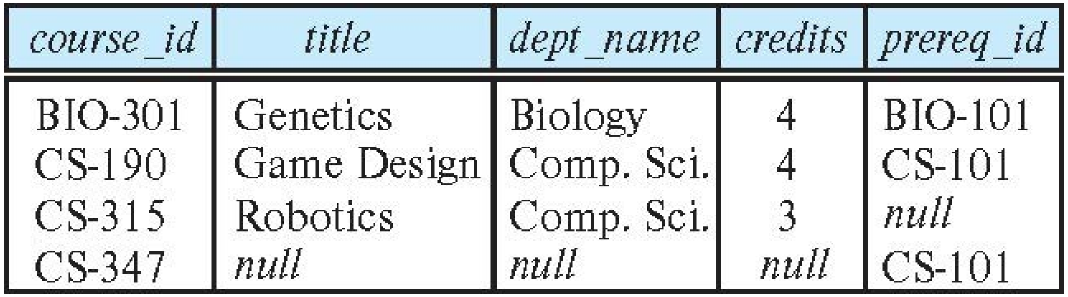


course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-347	*null*	*null*	*null*	CS-101

- In relational algebra: *course* ⟖ *prereq*

Full Outer Join

- *course* **natural full outer join** *prereq*

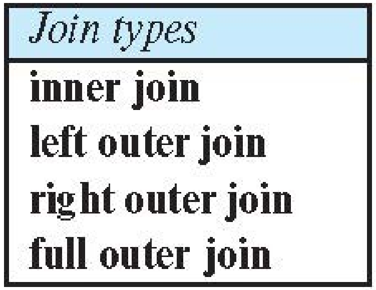


course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	*null*
CS-347	*null*	*null*	*null*	CS-101

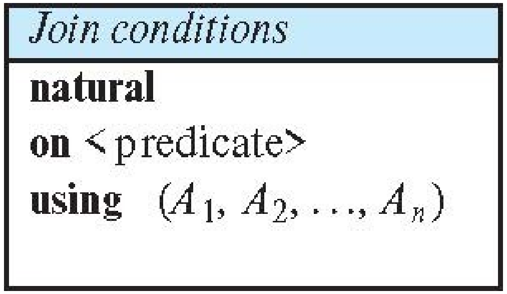
- In relational algebra: *course* ⟗ *prereq*

Joined Types and Conditions

- **Join operations** take two relations and return as a result another relation.
- These additional operations are typically used as subquery expressions in the **from** clause
- **Join condition** – defines which tuples in the two relations match.
- **Join type** – defines how tuples in each relation that do not match any tuple in the other relation (based on the join condition) are treated.



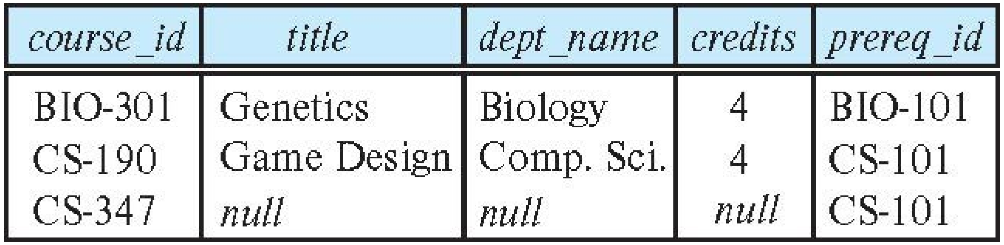
Join types
inner join
left outer join
right outer join
full outer join



Join conditions
natural
on < predicate>
using $(A_1, A_2, \ldots, A_n)$

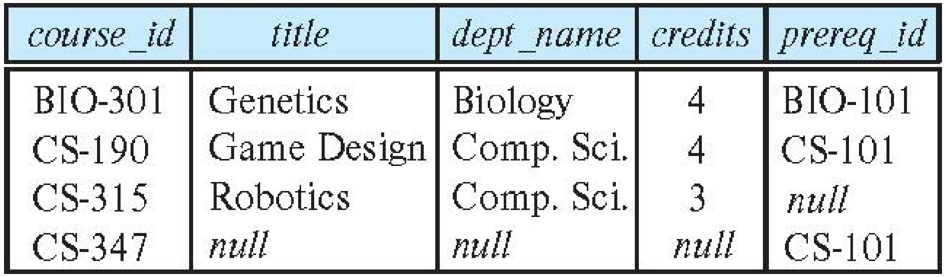
Joined Relations – Examples

- *course* **natural right outer join** *prereq*



course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-347	*null*	*null*	*null*	CS-101

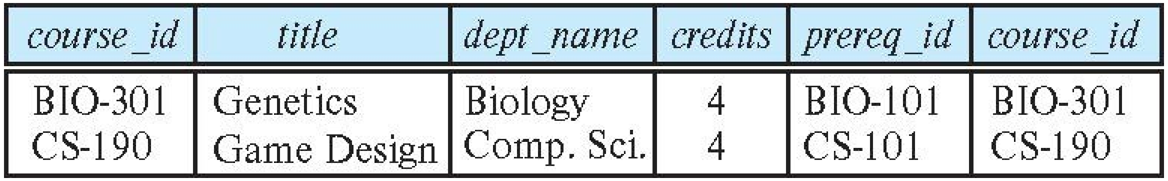
- *course* **full outer join** *prereq* **using** (*course_id*)



course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	*null*
CS-347	*null*	*null*	*null*	CS-101

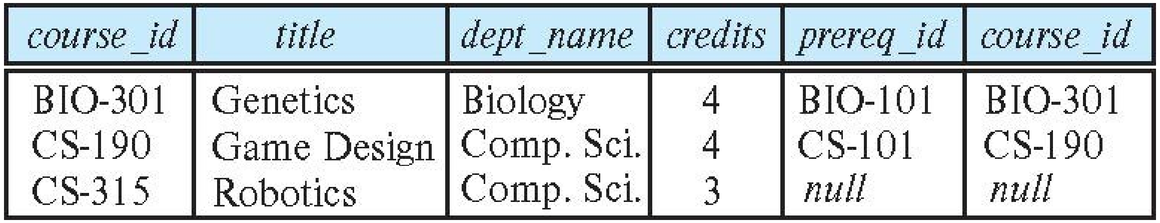
Joined Relations – Examples

- *course* **inner join** *prereq* **on**
  course.course_id = prereq.course_id



course_id	title	dept_name	credits	prereq_id	course_id
BIO-301	Genetics	Biology	4	BIO-101	BIO-301
CS-190	Game Design	Comp. Sci.	4	CS-101	CS-190

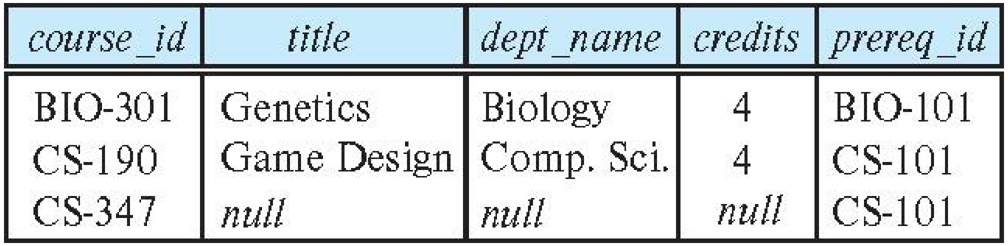
- What is the difference between the above, and a natural join?
- *course* **left outer join** *prereq* **on**
  course.course_id = prereq.course_id



course_id	title	dept_name	credits	prereq_id	course_id
BIO-301	Genetics	Biology	4	BIO-101	BIO-301
CS-190	Game Design	Comp. Sci.	4	CS-101	CS-190
CS-315	Robotics	Comp. Sci.	3	*null*	*null*

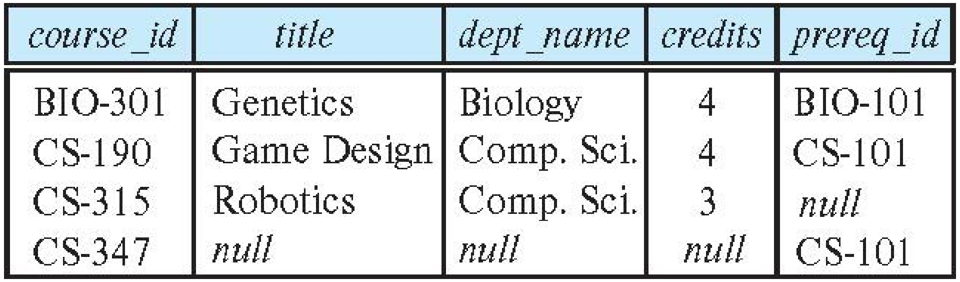
Joined Relations – Examples

- *course* **natural right outer join** *prereq*



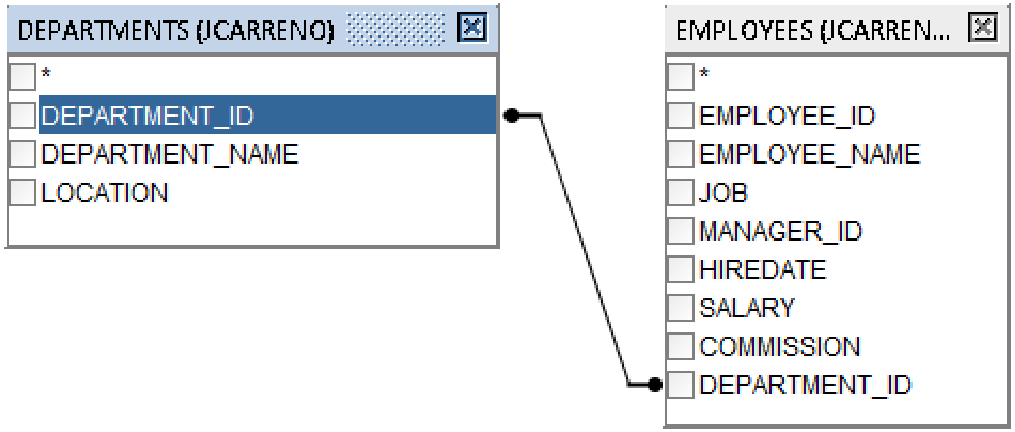
course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-347	*null*	*null*	*null*	CS-101

- *course* **full outer join** *prereq* **using** (*course_id*)



course_id	*title*	*dept_name*	*credits*	*prereq_id*
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	*null*
CS-347	*null*	*null*	*null*	CS-101

Oracle Example



DEPARTMENTS (JCARRENO) ☒

□ *

□ DEPARTMENT_ID

□ DEPARTMENT_NAME

□ LOCATION

EMPLOYEES (JCARREN... ☒

□ *

□ EMPLOYEE_ID

□ EMPLOYEE_NAME

□ JOB

□ MANAGER_ID

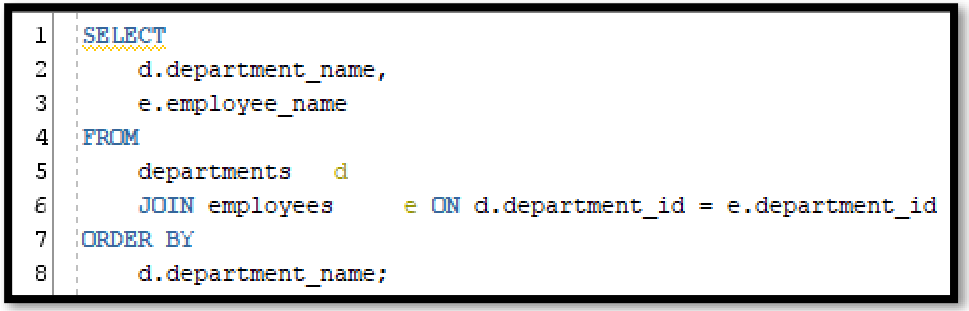
□ HIREDATE

□ SALARY

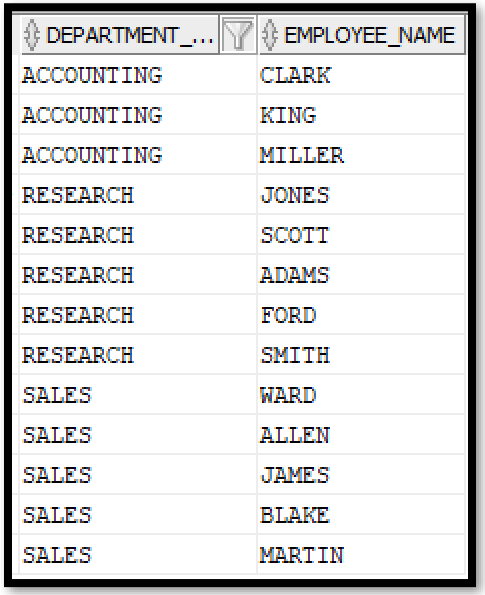
□ COMMISSION

□ DEPARTMENT_ID

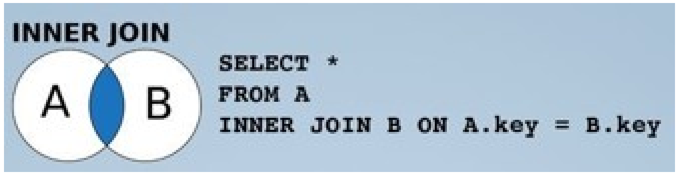
Oracle Example



```
SELECT
    d.department_name,
    e.employee_name
FROM
    departments   d
    JOIN employees     e ON d.department_id = e.department_id
ORDER BY
    d.department_name;
```

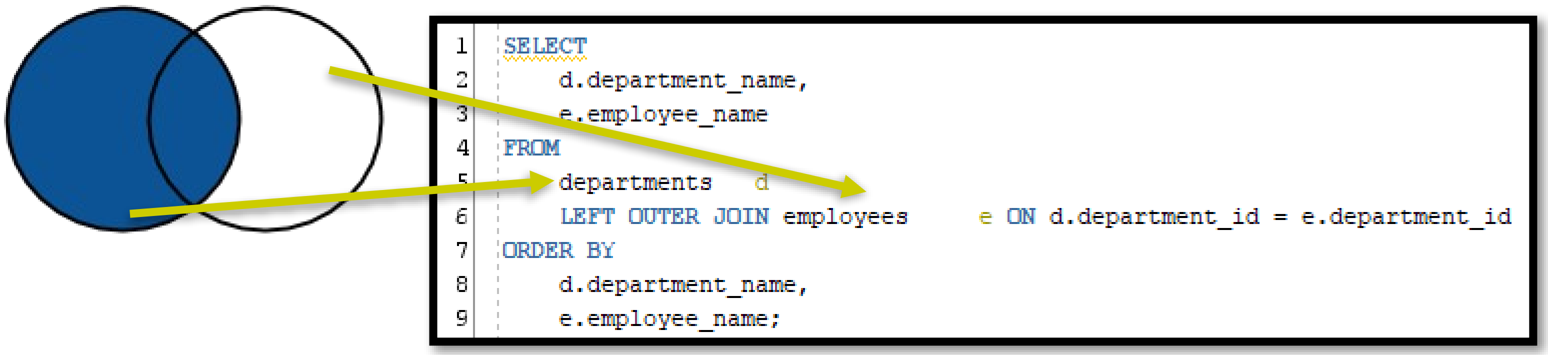


DEPARTMENT_...	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	SMITH
SALES	WARD
SALES	ALLEN
SALES	JAMES
SALES	BLAKE
SALES	MARTIN

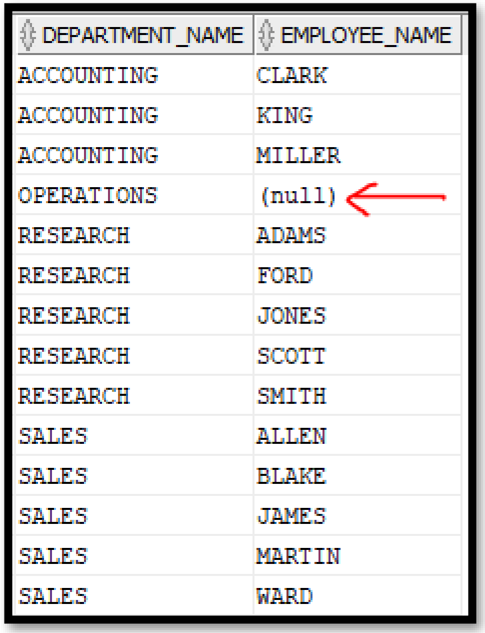


```
SELECT *
FROM A
INNER JOIN B ON A.key = B.key
```

Oracle LEFT [OUTER] JOIN

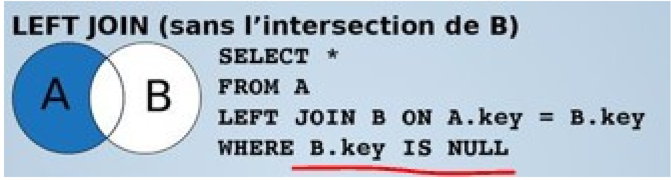


```
SELECT
    d.department_name,
    e.employee_name
FROM
    departments    d
    LEFT OUTER JOIN employees    e ON d.department_id = e.department_id
ORDER BY
    d.department_name,
    e.employee_name;
```



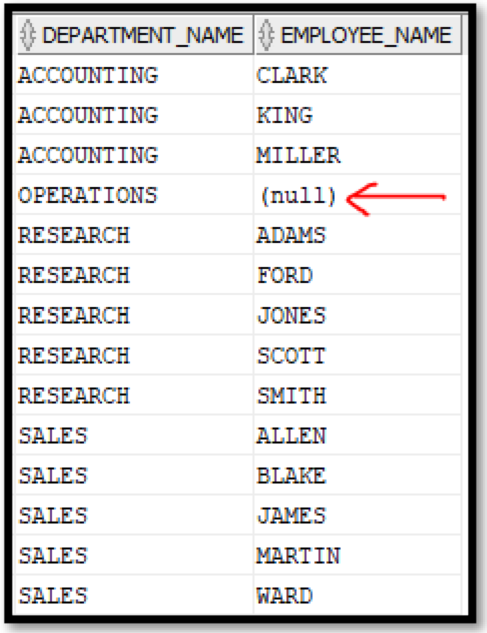
DEPARTMENT_NAME	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
OPERATIONS	(null)
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	SMITH
SALES	ALLEN
SALES	BLAKE
SALES	JAMES
SALES	MARTIN
SALES	WARD

Oracle LEFT [OUTER] JOIN


LEFT JOIN (sans l'intersection de B)

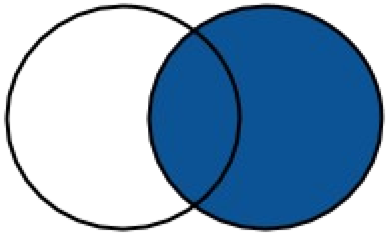
```
SELECT *
FROM A
LEFT JOIN B ON A.key = B.key
WHERE B.key IS NULL
```

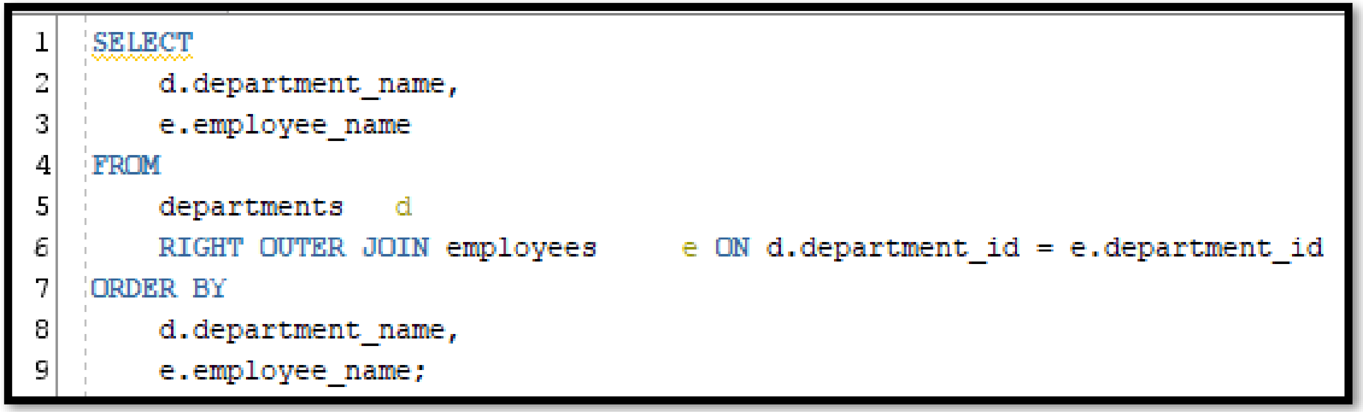
Y como obtengo empleados que no están asignados a un departamento?



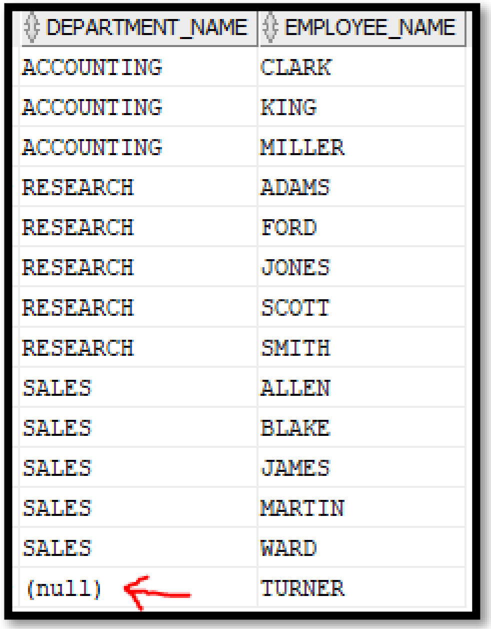
DEPARTMENT_NAME	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
OPERATIONS	(null) ←
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	SMITH
SALES	ALLEN
SALES	BLAKE
SALES	JAMES
SALES	MARTIN
SALES	WARD

Oracle RIGHT [OUTER] JOIN



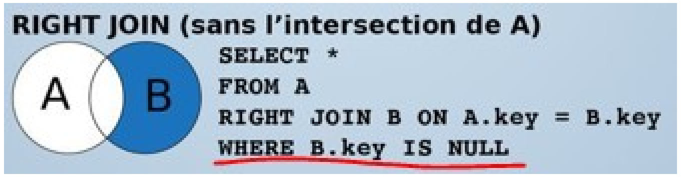


```
SELECT
    d.department_name,
    e.employee_name
FROM
    departments   d
    RIGHT OUTER JOIN employees     e ON d.department_id = e.department_id
ORDER BY
    d.department_name,
    e.employee_name;
```



DEPARTMENT_NAME	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	SMITH
SALES	ALLEN
SALES	BLAKE
SALES	JAMES
SALES	MARTIN
SALES	WARD
(null) ←	TURNER

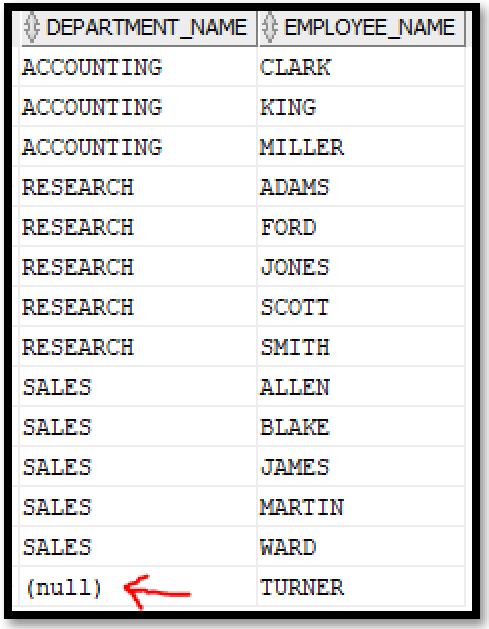
Oracle RIGHT [OUTER] JOIN



RIGHT JOIN (sans l'intersection de A)

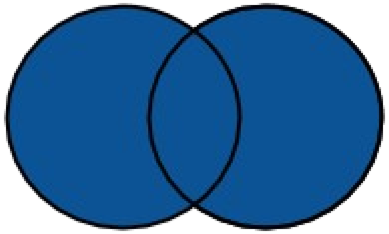
```
SELECT *
FROM A
RIGHT JOIN B ON A.key = B.key
WHERE B.key IS NULL
```

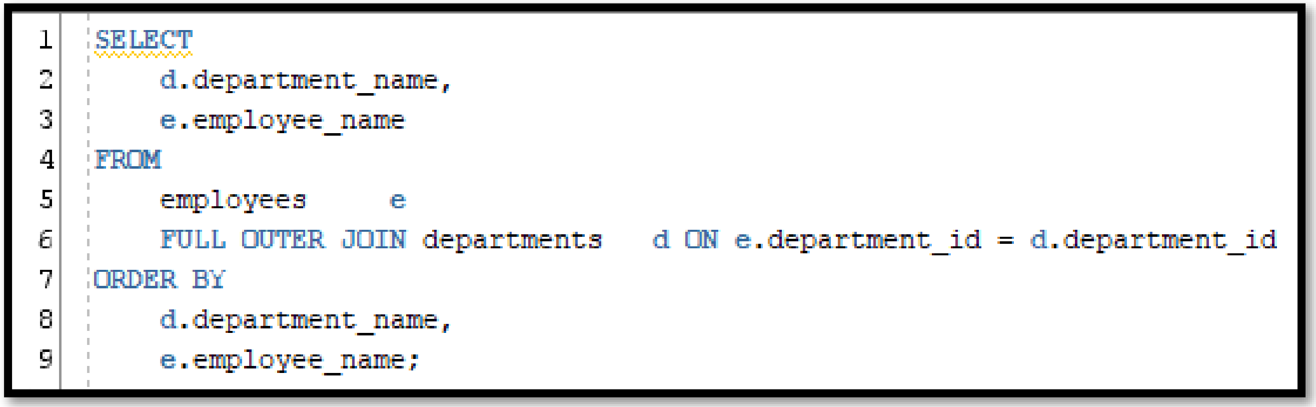
Y como obtengo los departamentos que no tienen empleados?



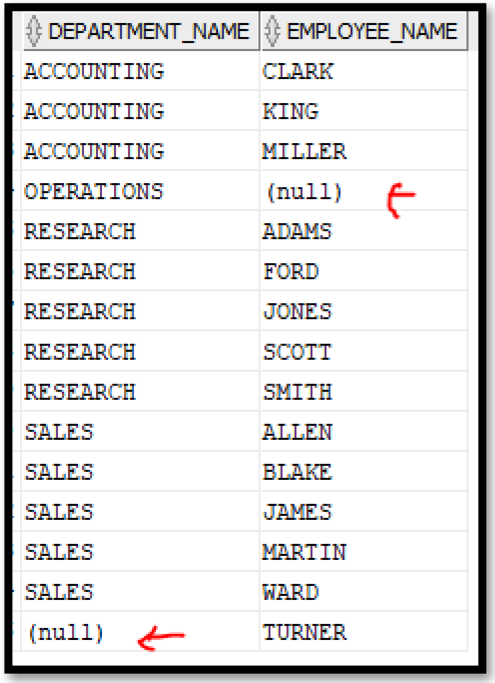
DEPARTMENT_NAME	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	SMITH
SALES	ALLEN
SALES	BLAKE
SALES	JAMES
SALES	MARTIN
SALES	WARD
(null)	TURNER

Oracle FULL [OUTER] JOIN



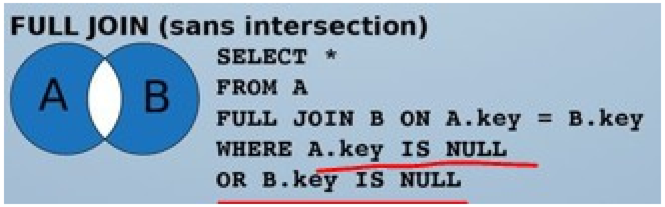


```
SELECT
    d.department_name,
    e.employee_name
FROM
    employees    e
    FULL OUTER JOIN departments   d ON e.department_id = d.department_id
ORDER BY
    d.department_name,
    e.employee_name;
```



DEPARTMENT_NAME	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
OPERATIONS	(null)
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	SMITH
SALES	ALLEN
SALES	BLAKE
SALES	JAMES
SALES	MARTIN
SALES	WARD
(null)	TURNER

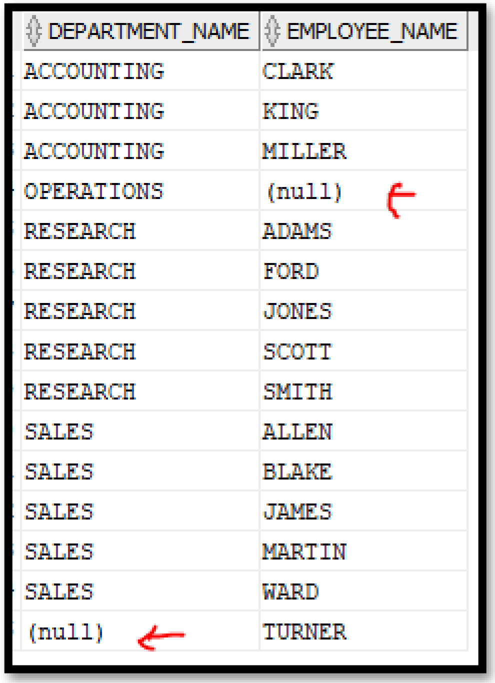
Oracle FULL [OUTER] JOIN



FULL JOIN (sans intersection)

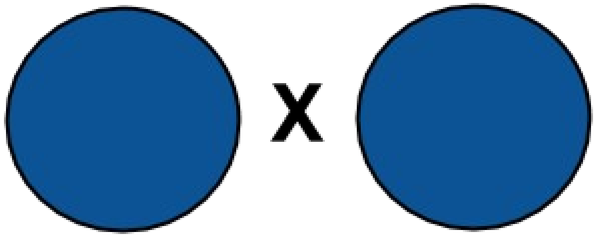
```
SELECT *
FROM A
FULL JOIN B ON A.key = B.key
WHERE A.key IS NULL
OR B.key IS NULL
```

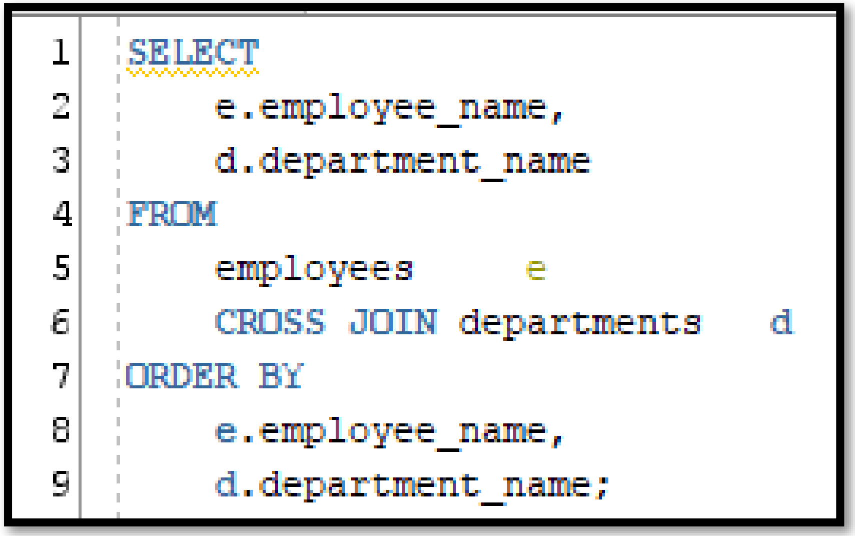
Y como obtengo los departamentos que no tienen empleados y empleados sin departamento?



DEPARTMENT_NAME	EMPLOYEE_NAME
ACCOUNTING	CLARK
ACCOUNTING	KING
ACCOUNTING	MILLER
OPERATIONS	(null) ←
RESEARCH	ADAMS
RESEARCH	FORD
RESEARCH	JONES
RESEARCH	SCOTT
RESEARCH	SMITH
SALES	ALLEN
SALES	BLAKE
SALES	JAMES
SALES	MARTIN
SALES	WARD
(null) ←	TURNER

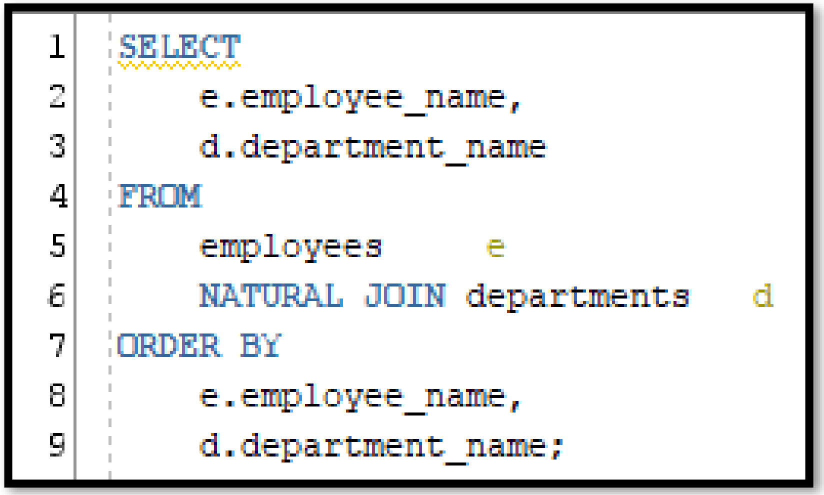
Oracle Cross join





```
SELECT
    e.employee_name,
    d.department_name
FROM
    employees     e
    CROSS JOIN departments   d
ORDER BY
    e.employee_name,
    d.department_name;
```

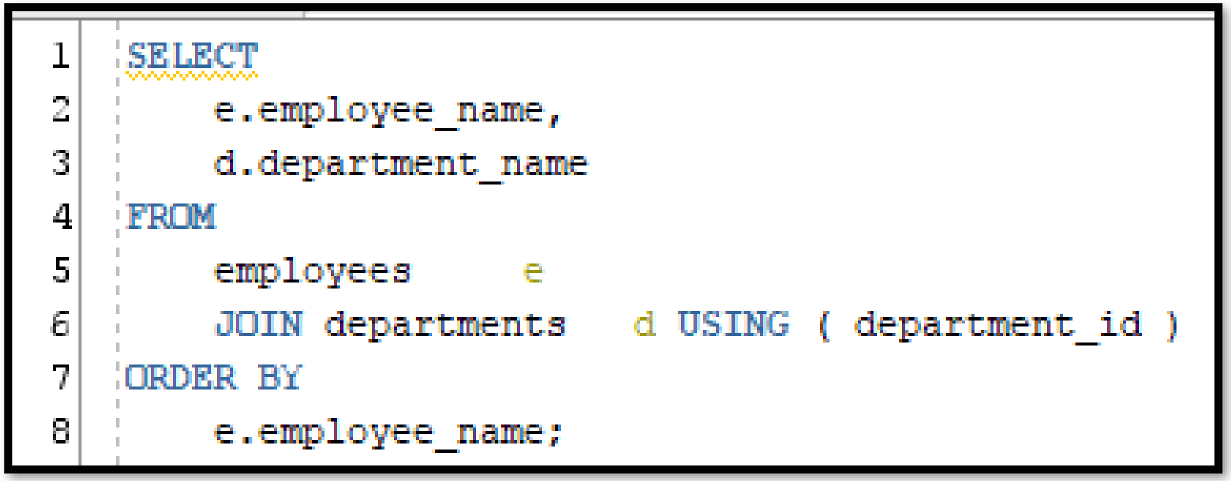
Oracle Natural Join



```
SELECT
    e.employee_name,
    d.department_name
FROM
    employees     e
    NATURAL JOIN departments    d
ORDER BY
    e.employee_name,
    d.department_name;
```

You can't apply any aliased filters to columns used in natural joins

Oracle Using



```
SELECT
    e.employee_name,
    d.department_name
FROM
    employees     e
    JOIN departments   d USING ( department_id )
ORDER BY
    e.employee_name;
```

You can't apply any aliased filters to columns used in natural joins